MyUI: Generating Accessible User Interfaces from Multimodal Design Patterns

Matthias Peissner, Dagmar Häbe, Doris Janssen, and Thomas Sellner
Fraunhofer Institute for Industrial Engineering IAO
Nobelstr. 12, 70569 Stuttgart, Germany
{matthias.peissner, dagmar.haebe, doris.janssen, thomas.sellner}@iao.fraunhofer.de

ABSTRACT
Adaptive user interfaces can make technology more accessible. Quite a number of conceptual and technical approaches have been proposed for adaptations to diverse user needs, multiple devices or multiple environments. Little work, however, has been directed at integrating all the essential aspects of adaptive user interfaces for accessibility in one system. In this paper, we present our generic MyUI infrastructure for increased accessibility through automatically generated adaptive user interfaces. The multimodal design patterns repository serves as the basis for a modular approach to individualized user interfaces. This open and extensible pattern repository makes the adaptation rules transparent for designers and developers who can contribute to the repository by sharing their knowledge about accessible design. The adaptation architecture and procedures enable user interface generation and dynamic adaptations during run-time. For the specification of an abstract user interface model, a novel statecharts-based notation has been developed. A development tool supports the interactive creation of the graphical user interface model.

Author Keywords
Adaptive user interfaces; design patterns; accessibility

ACM Classification Keywords
D.2.11 [Software Engineering]: Software Architectures – Patterns; H.5.2 [Information Interfaces and Presentation]: User Interfaces – User-centered design.

General Terms
Design; Human Factors.

INTRODUCTION

Adaptive user interfaces have been widely recognized as a promising means towards accessible technology (e.g. [11] [29] [30] [35]). Identifying individual and situational user needs and providing dynamically personalized user interfaces can overcome significant barriers of use. Despite the big potentials and the long research tradition of adaptive user interfaces, current approaches are still far from being adopted in the market. We argue that effective approaches to adaptive user interfaces for increased accessibility must satisfy a couple of requirements to gain significant attention beyond the scientific community:

They will require a conceptual and technical framework which allows for *extensive adaptations* to cover diverse user capabilities and needs, devices and contexts of use. Approaches that address only certain types of adaptations or disabilities and approaches for cross-platform interfaces may have their justification for other specific purposes or only parts of the problem. But they will not overcome the challenge of providing universal access for the widest range of users, devices and contexts of use.

Adaptations to such an extent require a huge knowledge base or extremely complex adaptation algorithms. It will hardly be possible to provide this in a monolithic system. *Modularity* will be needed to manage the complexity. For practical reasons, an *extensible* approach is regarded important to support starting with a manageable subset of design solutions and successive extensions.

Many users have problems with customization dialogues [15]. For users with disabilities and lower levels of ICT literacy the need for customization will be a significant barrier. Therefore, personalization which aims at increasing accessibility must include system-initiated adaptations and self-learning mechanisms to identify individual needs. *Adaptations during use* [6] can support an immediate adaptation to newly available knowledge about the user and the environment and can overcome problems of use directly when they occur. This will also help to cover altering capabilities in the course of aging and rehabilitation.

System-initiated adaptations during use can produce problems for the users. Confusing inconsistencies and the feeling of losing control can lead to bad usability and low acceptance [34]. Therefore, mechanisms to assure *transparency and controllability* of automatic interface adaptations are essential for the success of adaptive systems.

On the other hand, also developers and designers have reservations against generated and adaptive user interfaces. Main drawbacks include the relatively demanding work

EICS'12, June 25–26, 2012, Copenhagen, Denmark.

with abstract user interface specifications and the loss of control over the appearance of the generated user interface [15]. Hence, the popularity of adaptive systems in industrial software development will depend on *intuitive and efficient* approaches for the creation, *comprehensible* generation and adaptation mechanisms, and opportunities for *customizing* and modifying the resulting interfaces.

- Our MyUI system supports the further development and mainstreaming of accessible and adaptive user interfaces by addressing these above mentioned requirements. This paper makes the following contributions:
- We describe the MyUI framework and infrastructure to generate individualized user interfaces and perform adaptations to diverse user needs, devices and environmental conditions during run time. This approach relies on an extensible design patterns repository which includes adaptation rules and modular building blocks for user interface generation.
- We present mechanisms to increase the transparency and user control of system-initiated adaptations during use.
- We present MyUI development tools and a graphical user interface specification format for the efficient creation of adaptive applications.
- We evaluate our approach in preliminary studies from three perspectives: (1) effectiveness of the technical infrastructure, (2) usability and acceptability from an end user's point of view, and (3) initial feedback from developers.

RELATED WORK

Adaptation frameworks for accessibility

There have been a number of systems for generating and adapting user interfaces. Most of them concentrate on one specific purpose of adaptation: individual user needs and disabilities [5] [31], multiple devices and modalities, e.g. [26] [21] [2] or context conditions [12] [13]. SUPPLE automatically generates interfaces adapted to a person's device, preferences and physical abilities [11]. Adaptations in the SUPPLE system rely on optimizing a cost-function which includes all aspects to be considered for personalization. This cost function might provide opportunities for extensions to cover also perceptual and cognitive capabilities and environmental factors. However, already the generating of interfaces which reflect individual motor capabilities exceeds acceptable performance times for run time adaptations. Some frameworks have been proposed for extensive adaptations [8] [17]. But none of them has proven their practical feasibility in a complete implementation.

Motti [17] has recognized the importance of extensibility for generic adaptation frameworks. However, only little attention has been targeted towards open frameworks which explicitly aim at engaging other external experts to collaborate in the refinement and extension of the used adaptation mechanisms and rules.

Run-time adaptations and usability

Only few systems provide automatic run-time generation and adaptation (cf. [11] [23]). Nevertheless, a lot has been written about the potential problems of self-adaptive user interfaces, particularly the lack of transparency and the lack of controllability. However, mechanisms to overcome these shortcomings have received relatively little attention in the literature. In a recent paper, Dessart et al. describe a first attempt to improve the transparency of run-time adaptations by animated transitions. They propose a taxonomy of adaptation categories and suitable transitions [7]. For the problem of controllability, not much specific guidance is available. Findlater and McGrenere provide an overview of some empirical studies which compare adaptable and adaptive user interfaces. On the basis of their own study on personalized menus they conclude that a mixed-initiative design which combines adaptable and adaptive mechanisms in one system will be the best to satisfy diverse users [10]. MICA is a good example for a modern mixed-initiative system which recommends customizations but leaves the decision to the user [4]. However, adaptive user interfaces that aim at accessibility improvements might need approaches where the system takes a more proactive role.

Developing adaptive user interfaces

The growing need for user interfaces that run on multiple devices has motivated a lot work in the field of interface generation and abstract user interface descriptions. The most prominent approaches include MARIA [27] and the earlier TERESA [26] with ConcurTaskTrees [25], the Personal Universal Controller (PUC) [21] as used in Huddle [23] and Uniform [22], UIML [1] and canonical abstract prototypes (CAP) with recent modifications in CAP3 [33]. Trewin et al. present requirements for abstract languages to support accessibility. They examine four languages and conclude that extensions will be needed to meet the specific requirements of universal usability [32].

For the case of PUC, Nichols et al. have shown that user interface generation can increase the usability of interfaces for multiple devices [20]. However, support for branding and customization of generated user interfaces is a weak spot in PUC – and most other systems. In many systems for user interface generation, not only controlling the final appearance of the user interface is difficult but also understanding the connection between the specification and the final result [18]. Another frequently mentioned problem is the high threshold of learning a new language [18] – even if for many approaches a graphical editor is available to support the creation of an abstract user interface model. Finally, model-based tools require design activities at an unusually high level of abstraction. Gummy [16] and Damask [14] are notable attempts to better align model-based user interface development with current design

practice. Both tools allow developers working on a concrete design for one device as a starting point for later abstractions. Gummy builds a corresponding UIML description from the concrete design. Damask uses higher-level design patterns selected by the designer to generate designs for other devices which can then be modified if desired. Both tools are not ready to be used for the specification of accessible and adaptive interfaces. They are restricted to cross-device design. The pattern approach in Damask might allow for extensions. However, as the used patterns work on a very specific level it would be very difficult to extend the approach to accessibility in general. Moreover, Damask generates designs, not complete interfaces. User interface adaptations are not addressed.

OVERVIEW OF THE MYUI SYSTEM

MyUI provides individualized user interfaces which are accessible to a broad range of users. The conceptual framework of MyUI contributes to the further development in the field of adaptive user interfaces by focusing on the following aspects (cf. [28]):

- *Generic framework for manifold adaptations:* The MyUI technical framework allows for adaptations to diverse user needs, different devices[1] and changing context conditions. Therefore, MyUI user interfaces adapt their presentation formats and modalities, the interaction mechanisms and the navigation paths.
- *Modular, extensible and open:* A modular approach is taken to manage the huge amount of possible user interface solutions. For practical reasons, extensibility is important to support a quick start with a manageable subset of design solutions and later extensions. Modularity and extensibility are achieved by a design patterns approach to adaptive user interfaces. The MyUI design patterns repository is publically available [19]. Thus, the underlying adaptation rules can be reviewed, refined and extended by other experts in the field.
- *Self-learning and adapting during use:* The model for MyUI user interface adaptations is the gradual adaptation between two human communication partners. In the beginning, both show quite careful and neutral behaviors. Over time, they learn to know each other better and better. A mutual understanding is established and improved by perceiving the partner's feedback to one's own communication acts, e.g. explicit remarks, emotional facial expressions, etc. Without an explicit interview to capture personal preferences and attitudes, human communicators adapt to each other in a quick, natural and smooth way during the interaction. MyUI strives to imitate such a smooth and natural adaptation process in accessible and highly individualized user interfaces. For this purpose, the MyUI system is collecting information about the user during the interaction and updates the user profile accordingly. In order to cover dynamic user profile changes, the MyUI framework supports run-time rendering and run-time adaptations of the user interface.
- *Transparent and controllable:* To assure high levels of usability and user acceptance, MyUI provides mechanisms to help the users to recognize and understand user interface adaptations.
- *Development infrastructure:* The mainstreaming of accessible and adaptive user interfaces is supported by a set of tools to facilitate easy and efficient design and development processes.

GENERATING AND ADAPTING USER INTERFACES

MyUI adaptive user interfaces are generated and adapted in a three-stage process (see figure 1).

User Interface Parameterization

User interface parameterization is the first step in the MyUI user interface generation and adaptation process. The result (output) of this first step is the *MyUI User Interface Profile* which defines general characteristics of the user interface. Examples for variables include bodyTextFontSize, displayMode (with values from "text only" over "mainly text" etc. to "graphics only") and voiceInput (on/off). The settings of the user interface profile are valid throughout the entire user interface and in all interaction situations of an application. During user interface parameterization information from three different input sources is processed:

- Information about the currently available and used I/O devices from the *Device Profile.*
- Information about the user and the current environment from the *User Profile.*
- Customization settings as defined by the developer of a MyUI application in the *Customization Profile*.

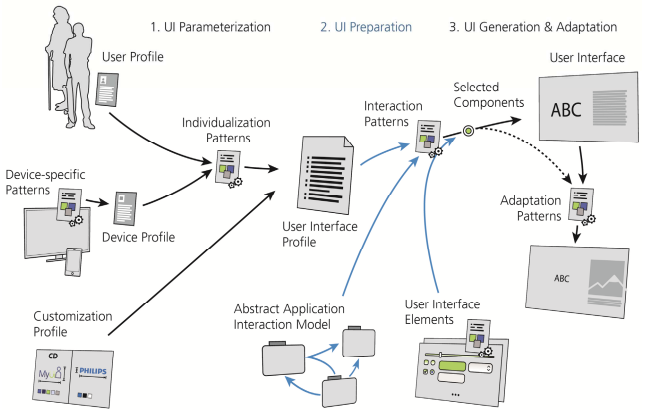

Figure 1 MyUI user interface generation and adaptation

[1] The current MyUI application prototypes run on a web-based iTV platform and iPhone. The MyUI technical framework, however, is generic and not restricted to iTV.

User interface parameterization can be regarded as a transformation of these profiles into the User Interface Profile. A *MyUI User Interface Profile* is initiated at the beginning of a new interaction session with a MyUI application. A repeated user interface parameterization (i.e. user interface profile update) is triggered when the available information about the user, the environment or the available devices changes significantly.

User Interface Preparation

User interface preparation denotes the process of selecting the most suitable user interface components and elements for the current situation. The major input of this process step includes

- the *Abstract Application Interaction Model* (AAIM) which defines active interaction situations for each state of the application and
- the *User Interface Profile* which reflects the specific requirements related to the current user, environment and device setup (see above).

User interface preparation is triggered every time when a new state in the AAIM is entered or when changes have occurred to the user interface profile. The AAIM is described in later sections of this paper.

User Interface Generation and Adaptation

The selected user interface components are rendered to an individual user interface. The major input to this step is the set of selected user interface components as a result of the user interface preparation. The output is a complete user interface which is consistent with the currently available knowledge about individual user needs and context requirements at any time during the interaction.

Dynamic and system-initiated user interface adaptations during run time are a main feature of the MyUI system. Run-time adaptations can be considered as a repeated user interface generation with mechanisms to switch from one instance of a user interface to another instance. In summary, the last step in the MyUI user interface generation and adaptation process includes the following three activities which all take place during run-time:

- *User interface generation*: At the beginning of a new interaction session, a complete user interface is created and rendered on the basis of the selected components (result of user interface preparation).
- *Profile updates:* In a permanent process during an interaction session, relevant events from the interaction are fed back to the Context Manager to update the user profile and device profile. Profile updates trigger the entire three-stage adaptation process again.
- *User interface adaptations during use:* When new user interface components and elements have been selected in a repeated user interface preparation process, user interface adaptations are triggered and executed.

MYUI DESIGN PATTERNS

Design patterns as modular building blocks

To cover the great heterogeneity of users, environments and devices, MyUI follows a modular approach to user interface development which relies on multimodal user interface design patterns. The MyUI design patterns contain the knowledge needed to perform the described three-stage process of user interface generation and adaptation. Individual accessibility is achieved by composing design patterns which provide proven solutions for specific interaction situations and characteristics of the user, environment and device. Adapting the user interface means switching from one design pattern of a bundle (e.g. all patterns for single selection from a list of options) to another pattern of the same bundle which is hypothesized to be the most appropriate for the current context.

MyUI Design Patterns Repository

The MyUI patterns repository includes the design patterns in a human-readable description format. The repository is maintained as a publicly available media wiki (see figure 2) [19]. Hence, the entire body of knowledge and rules engaged in the MyUI user interface generation and adaptation is easily accessible and transparent to everyone. Each pattern is described in a defined structure as proposed by Borchers [3] and related to other patterns of different types and levels of abstraction. Each pattern is linked to a reusable software component and associated with a source code representation of the described solution to put the recommended guideline into action in the MyUI adaptive user interface.

Extensibility is an important aspect of the design patterns repository. The current body of design patterns reflects the interaction requirements of initial MyUI demo applications, i.e. email service and instant messaging. For future applications, additional patterns will be needed. These can be simply added to the pattern repository without requiring a new version of a monolithic patterns document. Also new knowledge about the current patterns can lead to modifications and refinements. The human-readable and

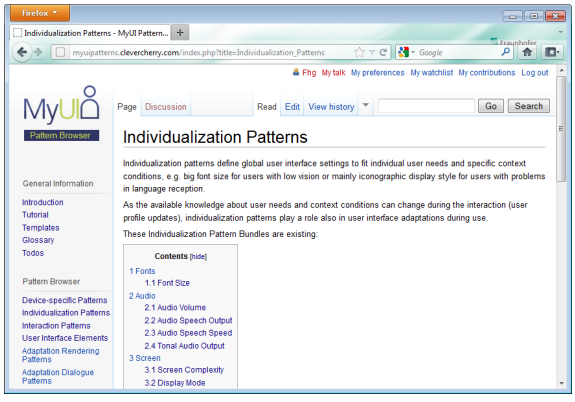

Figure 2 The MyUI Pattern Browser is the developer's access to the MyUI Pattern Repository

easy format, the well-known wiki platform and the little effort for changes lead to a low barrier for the community to use, discuss and improve the design patterns. With both, a machine-readable and a human-readable part, the MyUI design patterns repository bridges the gap between guidelines for accessible design and generative user interfaces.

Types of design patterns

The MyUI design patterns repository includes different categories of patterns. Each pattern type fulfills distinct functions in the MyUI adaptation framework (see figure 1):

Device-specific patterns create the device profile on the basis of primitive device features as provided by the used I/O devices. The device profile variables prepare the step of user interface parameterization by providing device-specific ranges of user interface settings from which individualization patterns select the most suitable (e.g. potential font sizes). Changes in the device setup will lead immediately to updating the device profile.

Individualization patterns define global user interface settings to fit individual user needs and environmental conditions, e.g. big font size for users with low vision or iconographic presentation for users with problems in language reception. As the available knowledge about the user and environment can change during the interaction, individualization patterns play an important role for user interface adaptations during use.

These first two pattern types work on a general level of the user interface. Together, they are responsible for setting and adjusting global variables in the user interface profile. An additional input to the user interface profile comes from the customization profile which allows for corporate-, project- or brand-specific customizations, e.g. colors or font styles. The customization settings are used in the user interface profile in a 1:1-manner without transformations. Therefore, no specific patterns are needed for customization.

Interaction patterns provide suitable user interface components for a current interaction situation as specified in the AAIM, e.g. a list element for an interaction situation in which a user can select from a set of options. For each interaction situation, a bundle of different interaction design patterns exists. They differ in appearance or interaction modality to support different user needs and context conditions. The selection of the best suitable interaction pattern from a current bundle is done on the basis of specific variables of the user interface profile.

User interface elements are the building blocks for the interaction patterns. While interaction patterns can be regarded as components to support a given interaction situation, the user interface elements provide generic primitives required to compose the interaction patterns.

Adaptation patterns cover the dynamics of the adaptation processes. They define the mechanisms of switching from one instance of a user interface to another. Adaptation patterns are described in more detail in a later section.

MyUI Design Patterns Language

According to Borchers' understanding, patterns are not isolated but refer to other patterns. In a hierarchical pattern language, larger patterns refer to smaller-scale patterns for the solution they describe. And smaller-scale patterns can only be used in a certain type of context which is the result of applying larger-scale patterns [3]. In the MyUI design pattern language, references between patterns play an important role. For a systematic use of references a classification of relations between patterns has been defined. Table 1 provides an overview of the relations used in the MyUI pattern language.

Table 1 Relations between MyUI design patterns

Relation	Description
A substitutes B	Both patterns A and B serve the same purpose in the MyUI framework and both patterns can never be active at the same time. This relation is used to create bundles of related patterns which support different user needs or context conditions.
A requires B	Pattern B describes parts of the higher-level solution addressed by pattern A. This relation is used to structure the design patterns repository in a vertical way. It links higher-level patterns to lower-level patterns for a detailed description of the solution.
A is required by B	This is the inverse relation of requires.
A sets <variables> as required by B	Patterns of bundle A set one or more global MyUI variables which specify (parts of) the solutions as provided by the patterns of bundle B.
A requires <variables> as set by B	This is the inverse relation of sets <variables> as required by.
A sets <variables> as used by B	Patterns of A set one or more global MyUI variables which are used to select the most suitable pattern from bundle B.
A uses <variables> as set by B	This is the inverse relation of sets <variables> as used by.

ADAPTATION PATTERNS FOR TRANSPARENCY AND CONTROLLABILITY

User interface adaptations are executed by the MyUI adaptation engine. The adaptation engine recognizes mismatches of the user interface components currently displayed and the components currently selected by the user interface preparation process. A mismatch triggers a run-

time adaptation. Adaptations are performed by two types of MyUI adaptation patterns:

- *Adaptation rendering patterns* specify the graphical rendering process to smoothly but obviously switch from one user interface instance to another, e.g. animated transitions to grow small fonts to bigger fonts.
- *Adaptation dialogue patterns* specify the interaction dialogue which takes place around the actual adaptation to make sure that the user is aware of the adaptation and can control the system's adaptation behavior.

Adaptation rendering patterns

MyUI adaptation rendering patterns make extensive use of animations. They support orientation by creating continuity between the user interface before and after the adaptation. Animated transitions shall draw the end user's attention to the screen areas where adaptations occur and shall help them to understand that and how the new user interface is a modification of the former user interface. When, for example, an increased font size results in hiding menu options which were directly available before the adaptation, an animation can communicate to the user that the hidden options are now available via the »more« button.

Adaptation dialogue patterns

Adaptation dialogue patterns specify the dialogue between the user interface and the MyUI user in the course of an adaptation. This dialogue typically includes a notification and interaction options for the user to influence the system's adaptation behavior. In MyUI, two types of adaptation dialogue patterns are distinguished:

- *System-initiated adaptation dialogue patterns* are triggered by the system.
- *User-initiated adaptation dialogue patterns* describe customization dialogues in which the user modifies the user profile or the user interface profile.

The latter make MyUI a mixed-initiative system with adaptive and adaptable components. More interesting, however, are system-initiated adaptation dialogue patterns. They aim at increasing the usability and acceptability by making system-initiated adaptations more transparent and controllable. The MyUI patterns repository includes the following system-initiated adaptation dialogue patterns.´

Explicit Confirmation before Adaptation

Before performing the adaptation the system requests the user to accept or reject the adaptation. The user's decision is supported by a preview of the adaptation effect (figure 3). If the user rejects the adaptation, the dialogue box is closed and the user interface is not changed. If the user accepts the adaptation or if the system receives no user input (time-out), the dialogue box is closed and the adaptation is carried out. The MyUI context management infrastructure is informed about the user's decision to refine the user profile.

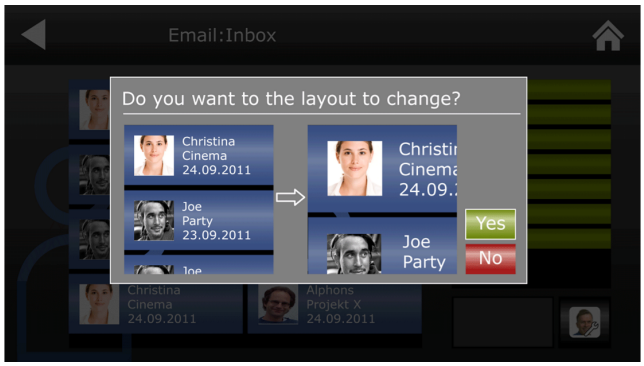

Figure 3 Explicit Confirmation before Adaptation: Dialogue box with preview (UI concept sketch)

Explicit Confirmation after Adaptation

The adaptation is triggered and performed automatically. After the adaptation a dialogue box asks the user if the changes shall be kept or undone. If the user rejects, the adaptation is undone. If the user accepts or a time-out event is recognized, the adaptation is kept. The user's decision is fed back to the context management infrastructure.

Automatic Adaptation with Implicit Confirmation

The adaptation is triggered and performed automatically. While rendering the adaptation, the system provides an icon-based notification in a dedicated adaptation area. Moreover, the adaptation area offers buttons to undo the adaptation and to access to the user interface profile and the user profile. Figure 4 shows a sequence of three screens before, during and after an automatic adaptation with implicit confirmation.

(1) Before adaptation Permanent access to user profile and user interface profile via adaptation area (bottom right)

(2) During adaptation Pulsing icon (here chameleon) indicates on-going adaptation.

(3) After adaptation The user can undo the adaptation via button with curved backwards arrow.

Figure 4 Automatic adaptation with implicit confirmation

DEVELOPING AN ADAPTIVE APPLICATION WITH MYUI

MyUI provides a development toolkit which supports developers in the creation of adaptive applications. It is implemented as a plugin for the well-established Eclipse platform to facilitate the later integration into industrial settings. It supports creating, customizing, previewing and simulating adaptive applications. In addition, it provides access to tutorials, design patterns and information about the pattern-based adaptation process.

Automatically generating user interfaces changes the application developers' role significantly. Their impact on the specific user interface appearance is minimized in favor of an automated and rule-based user interface generation process which includes the provision of different user interfaces for users with different needs. The developers' main responsibility in MyUI is defining the functionality and application logic in an AAIM.

Abstract Application Interaction Model (AAIM)

The AAIM describes the interaction between the user and the application in a way which is independent of a specific appearance and concrete interaction mechanisms. It serves as a basis for the generated and adapted user interfaces by defining the common ground of all possible user interfaces. The MyUI AAIM extends the UML 2 State Machine Diagram [24]. Statecharts allow modeling the interaction without going into details about the presentation modalities or used control elements. These aspects are subject to adaptations in MyUI and therefore not part of the AAIM specifications. Figure 5 presents a part of the AAIM for the MyUI email application.

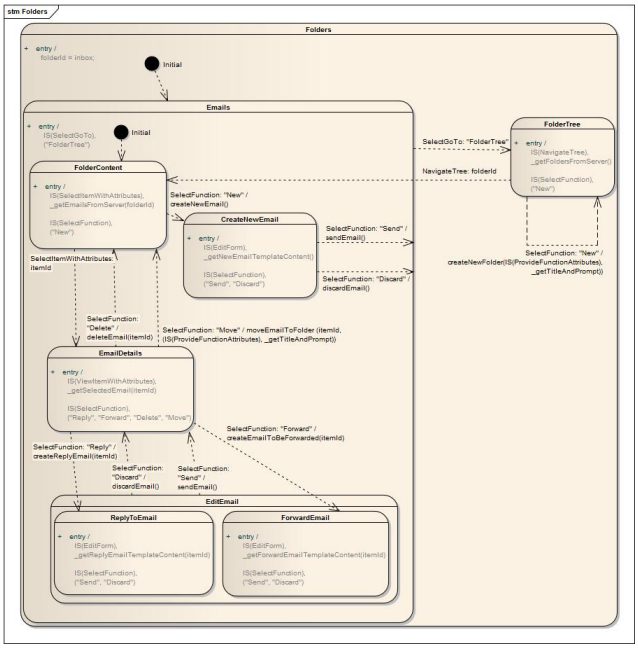

Figure 5 AAIM of the MyUI email application (excerpt)

Interaction situations as basis for adapting the user interface

The concept of interaction situations is essential for the MyUI adaptation framework. An interaction situation is an abstract super-set of user interface components and controls which serve the same interaction purpose. Interaction situations represent the interaction options a user has at a certain point in the application. Interaction options include all activities a user can perform in the application, e.g. perceiving information, providing specific input, selecting options, etc. Typically, more than one user interface component or element can be used in one certain interaction situation, e.g. selecting a single item from a set of items can be supported by a selection list, a drop-down list or even an audio menu. This flexibility is an essential basis for user interface adaptations in MyUI. Depending on individual user needs, the most suitable interaction pattern can be selected for a given interaction situation. The different variants of user interface components for a common purpose are referred to as bundles of interaction patterns.

Interaction situations in states and transitions

Every state includes one or more interaction situations. Interaction situations apply to the state in which they appear. Interaction situations in a composite state refer to all sub-states of the composite state and persist as long as one of the sub-states is active. This can be used also for the definition of generic navigation options which are available throughout an entire application or even across different applications, e.g. home navigation and back navigation.

An interaction situation of a state can be associated with respective transitions from the state to other states. An interaction situation, for example, which provides a set of functions from which the user can select in the current state, is always associated with a set of transitions from the state to other states in which the triggered functions are executed.

Interaction situations can appear in states and as arguments of transitions between states of the AAIM. Interaction situations at transitions are needed to retrieve confirmation or additional information from the user needed to perform the transition and to enter the next state. As an example, some interaction situations SelectFunction require arguments for executing the selected function, e.g. the destination folder of the "move to folder" function or an explicit user confirmation before the execution of an irreversible action. These arguments are then collected by a separate interaction situation, e.g. a dialogue box which requests the needed information from the user.

Development Toolkit

The main view of the MyUI Development Toolkit (see figure 6) in Eclipse includes the MyUI Editor (right side), a project explorer and a list of available interaction situations (upper and lower left side).

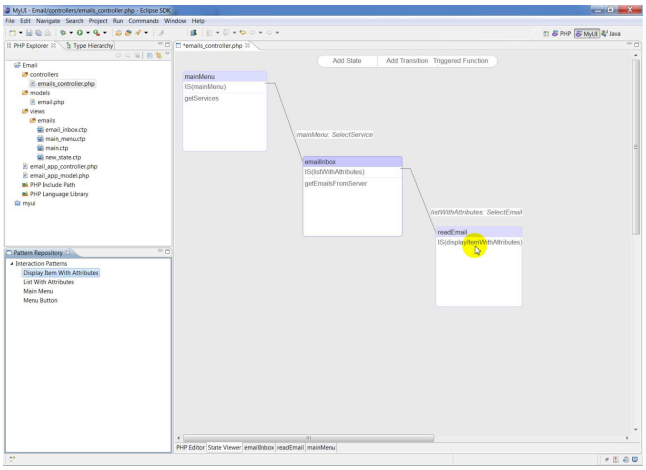

Figure 6 MyUI Development Toolkit

The MyUI Editor supports the easy creation of an AAIM by providing a graphical model view. A wizard helps to build the application skeleton by specifying general application properties. As a result, the generated file and folder structure of the application appears in the project explorer. The developer can then create new states in the model editor and add interaction situations by drag-and-drop from the list view in the lower left area. Thereby, specific parameters such as references to data sources can be entered. The related bundle of interaction patterns appears in the state. Associated transitions are automatically generated and can be connected to other existing states. In this way, the entire adaptive application can be successively modeled. The MyUI Editor also allows switching between the model view and the source code view. The designer can chose to create and refine the model in any of both views which are kept consistent.

Another component of the development toolkit is the MyUI pattern browser. Integrated into the Eclipse-based development toolkit, the pattern browser provides direct access to the MyUI pattern repository with detailed information about the design patterns and their roles within the generation and adaptation process. Thus, the developer can easily explore the available patterns and identify the needed interaction situations.

In order to support a deep understanding of the relationships between the abstract model and the resulting interfaces for different users and contexts, the development toolkit has integrated preview functionality. The developer can select and load user profiles and see how the generated user interfaces would look for these specific users.

Finally, the development toolkit offers some opportunities for customizing the generated user interface. Simulation facilities make it possible to validate the feasibility of customized font styles, color schemes, etc. for disabled users by providing a what-you-see-is-what-others-get view on the customized user interface.

EVALUATION

Framework validation

The practical feasibility of the technical framework has been evaluated in the development of a first adaptive iTV-based web application. This demonstrator includes a main menu for service selection and an email service.

The developed prototype covers extensive adaptations to individual perceptual, physical and cognitive capabilities and environmental conditions such as ambient noise and ambient light. Device-specific adaptations were not tested in this first cycle. The adaptation to individual user needs was tested by the use of two different personas with different levels of perceptual, physical and cognitive impairments. The functional tests yielded satisfying results. At any time, the MyUI system generated a meaningful and consistent user interface. Relevant updates in the user and context profile during the interaction always resulted in the expected run-time adaptations. The recorded system performance for user interface generation and adaptations during run-time was excellent. The time needed to generate a new instance of the user interface was equal to the (very short) time for loading a new page of the web application without adaptations.

User Studies

A couple of preliminary informal user studies were carried out to evaluate specific user interface solutions for users with certain limitations as documented in the MyUI design patterns. Especially, patterns which could not be substantiated by the literature were tested with older adults.

In an early paper prototype study with four participants (three female, one male, aged 70-89) we wanted to explore the ways in which older people respond to the concept of adaptive interfaces. The participants were presented a series of pages which set out representations of interfaces based on an interactive TV screen. The interface depicted was chosen mainly for its ability to demonstrate various forms of adaptation rather than as a representation of a proposed device. Simple tasks were assigned and according to selections made, the ‘screen’ layout changed. Participants were asked for their opinions on the changing layouts, and for any suggestions they wished to make. A main result of this study was that it was very hard to get feedback on the specific concept of adaptation. Older persons with low ICT literacy seem to have difficulties in understanding and recognizing system-initiated adaptations at all [9]. This finding challenges the overall concept of adaptive and accessible interfaces for a major target user group of MyUI.

Currently, we are preparing a user study to evaluate the proposed adaptation dialogue patterns. We assume that their effectiveness and acceptability depend on the subjective cost-benefit of the adaptation. Predominant costs might be associated with a higher wish for user control and therefore, a preference for explicit confirmations. But in situations where the benefits outweigh the costs, users might prefer

automatic adaptations because of their higher comfort of use. In our case, adaptations can cause further costs, e.g., when the adapted user interface requires additional interaction steps, e.g. increasing the font size requires extra scrolling. Benefits occur when an adaptation can increase the accessibility by eliminating a barrier of use. Besides these situational factors, also personal traits might influence the subjective cost-benefit ratio and the preference of one of the adaptation patterns. Users with lower ICT literacy or a higher need for security might tend to over-estimate the costs and therefore prefer explicit confirmations.

The current study will address the following questions:

- Which adaptation dialogue patterns are most effective in terms of transparency, controllability and acceptance?
- Which pattern is preferred by the users?
- Do these measures differ in different cost-benefit conditions and for different users?

The results will help us to design the mechanism for selecting the most appropriate adaptation pattern for different situations and user profiles.

Developers' feedback

Eight software developers participated in a preliminary study to collect feedback from the developers' perspective in a focus group setting. The participants were presented basic concepts of the MyUI project, a video which explains the work with the development toolkit and the AAIM of the email application. After a discussion and a round of questions and answers, the participants completed a questionnaire to evaluate the MyUI system on a number of Likert-scales. Figure 7 summarizes the results. Besides the relatively good overall assessment, a major finding was that some of the developers had severe problems in estimating the usefulness of the MyUI system. Due to the early stage in the development process the participants could not try a functional system but had to rely on the presented material. Therefore, it is not clear if these problems are due to the framework itself or to the presented material. Nevertheless, they pointed us to the importance of easy-to-understand instruction material to support an easy start with MyUI.

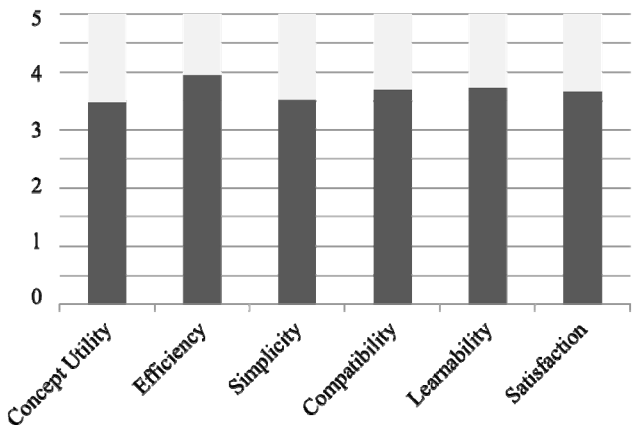

Figure 7 Initial developer feedback (mean ratings, n=8)

CONCLUSION

We have presented an overview of MyUI, a novel system which generates adaptive user interfaces to support the mainstreaming of accessibility. MyUI focuses on major requirements of adaptive interfaces for accessibility which have been neglected by previous systems in the field. MyUI provides a framework for extensive run-time adaptations to user characteristics, environmental conditions and used devices. Modularity and extensibility are achieved by a design patterns approach to adaptive user interfaces. Increasing the usability and acceptance of automatic run-time adaptations is a major topic in MyUI. Current user studies will help us to design transparent and controllable mechanisms. Finally, a statecharts-based approach to graphical user interface modeling with a dedicated toolbox aims at tackling current acceptance problems in the industrial software development.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union's Seventh Framework Program under grant FP7-ICT-248606. The authors acknowledge the help of other partners in the MyUI consortium in this work.

REFERENCES

1. Abrams, M., Phanouriou, C., Batongbacal, A.L., Williams, S. & Shuster, J. (1999), UIML: An Appliance-Independent XML User Interface Language. In Proc. WWW'8, Amsterdam: Elsevier, 1695-1708
2. Ali, M.F., Pérez-Quiñones, M.A., Abrams, M., & Shell, E. (2002). Building Multi-Platform User Interfaces with UIML. In Proceedings CADUI 2002, 255-266.
3. Borchers, J. O. (2001). A pattern approach to interactive design. Chichester, UK: John Wiley & Sons Ltd.
4. Bunt, A., Conati, C., & McGrenere, J. (2009). A Mixed-Initiative Approach to Interface Personalization. AI Magazine 30(4).
5. Coelho, J., Duarte, C., Biswas, P., & Langdon, P. (2011). Developing accessible TV applications. In Proceedings ASSETS '11. New York: ACM, 131-138.
6. Dieterich, H., Malinowski, U., Kühme, T. & Schneider-Hufschmidt, M. (1993). State of the Art in Adaptive User Interfaces. In: M. Schneider-Hufschmidt, T. Kühme & U. Malinowski (Eds.): Adaptive User Interfaces: Principles and practice. Amsterdam: North-Holland, 13–48.
7. Dessart, C.-E., Motti, V. G. & Vanderdonckt, J. (2011). Showing user interface adaptivity by animated transitions. In Proc. EICS '11. New York: ACM, 95-104.
8. Duarte, C. & Carriço, L (2006). A conceptual framework for developing adaptive multimodal

applications. In Proc. IUI '06. New York: ACM, 132-139.

9. Edlin-White, R., Cobb, S., Floyde, A., Lewthwaite, S., Wang, J. & Riedel, J. (2012). From guinea pigs to design partners - involving older people in technology design. In: P. Langdon, J. Clarkson, P. Robinson, J. Lazar & A. Heylighen (Eds.): Designing Inclusive Systems. London: Springer-Verlag, 155-164.
10. Findlater, L. & McGrenere, J. (2004) A comparison of static, adaptive, and adaptable menus. In Proceedings CHI'04, New York: ACM, 89-96.
11. Gajos, K. Z., Weld, D. S., & Wobbrock, J. O. (2010). Automatically generating personalized user interfaces with Supple. Artificial Intelligence 174, 12-13. 910-950.
12. Kane, S. K., Wobbrock, J. O. & Smith, I. E. (2008). Getting off the treadmill: evaluating walking user interfaces for mobile devices in public spaces. In Proceedings MobileHCI '08. New York: ACM, 109-118.
13. Lehmann, G., Blumendorf, M. & Albayrak, S. (2010). Development of context-adaptive applications on the basis of runtime user interface models. In Proceedings EICS '10. New York: ACM, 309-314.
14. Lin, J. & Landay, J. A. (2008). Employing patterns and layers for early-stage design and prototyping of cross-device user interfaces. In Proceedings CHI '08. New York: ACM, 1313-1322.
15. Mackay, W. E. (1991). Triggers and barriers to customizing software. In Proceedings CHI '91, New York: ACM, 153–160.
16. Meskens, J., Vermeulen, J., Luyten, K. & Coninx, K. (2008). Gummy for multi-platform user interface designs: shape me, multiply me, fix me, use me. In Proceedings AVI '08. New York: ACM, 233-240.
17. Motti, V. G. (2011). A computational framework for multi-dimensional context-aware adaptation. In Proceedings EICS '11. New York: ACM, 315-318.
18. Myers, B., Hudson, S. E. & Pausch, R. (2000). Past, present, and future of user interface software tools. ACM Trans. Comp.-Hum. Interact. 7, 1, 3-28.
19. MyUI Design Patterns Repository. Available at http://myuipatterns.clevercherry.com
20. Nichols, J., Chau, D. H., & Myers, B. A. (2007). Demonstrating the viability of automatically generated user interfaces. In Proceedings CHI '07. New York: ACM, 1283-1292.
21. Nichols, J & Myers, B. A. (2009). Creating a lightweight user interface description language: An overview and analysis of the personal universal controller project. ACM Trans. Comp.-Hum. Interact. 16, 4, 37 pages.
22. Nichols, J., Myers, B. A. & Rothrock, B. (2006). UNIFORM: automatically generating consistent remote control user interfaces. In Proceedings CHI '06. New York: ACM, 611-620.
23. Nichols, J., Rothrock, B., Chau, D. H. & Myers, B. A. (2006). Huddle: automatically generating interfaces for systems of multiple connected appliances. In Proceedings UIST '06. New York: ACM, 279-288.
24. Object Management Group (2011). Unified Modeling Language Superstructure, V 2.4.1, available at http://www.omg.org/spec/UML/2.4.1/Superstructure
25. Paterno, F. (1999). Model-Based Design and Evaluation of Interactive Applications. London: Springer-Verlag.
26. Paterno, F., Santoro, C., Mantyjarvi, J., Mori, G. & Sansone, S. (2008). Authoring pervasive multimodal user interfaces. Int. J. Web Eng. Technol. 4, 2, 235-261.
27. Paterno, F., Santoro, C. & Spano, L. D. (2009). MARIA: A universal, declarative, multiple abstraction-level language for service-oriented applications in ubiquitous environments. ACM Trans. Comp.-Hum. Interact. 16, 4, 30 pages.
28. Peissner, M., Schuller, A. & Spath, D. (2011). A design patterns approach to adaptive user interfaces for users with special needs. In Proceedings HCII'11, Berlin: Springer-Verlag, 268-277.
29. Ringbauer, B., Peissner, M., & Gemou, M. (2007). From "design for all" towards "design for one"– A modular user interface approach. In: C. Stephanidis (Ed.): Universal Access in HCI, Part I, HCII 2007, LNCS 4554, Berlin: Springer-Verlag, 517–526.
30. Savidis, A. & Stephanidis, C. (2004). Unified user interface design: Designing universally accessible interactions. Int. J. Interacting w. Comp. 16, 2, 243–270.
31. Stephanidis, C., Paramythis, A., Sfyrakis, M., Stergiou, A., Maou, N., Leventis, A., Paparoulis, G. & Karagiannidis, C. (1998). Adaptable and adaptive user interfaces for disabled users in the AVANTI project, in: IS&N 98 Proceedings, Springer-Verlag, 153-166.
32. Trewin, S., Zimmermann, G. & Vanderheiden, G. (2002). Abstract user interface representations: how well do they support universal access? SIGCAPH Comput. Phys. Handicap. 73-74 (June 2002), 77-84.
33. Van den Bergh, J., Luyten, K. & Coninx, K. (2011). CAP3: context-sensitive abstract user interface specification. In Proceedings EICS '11. New York: ACM, 31-40.
34. Weld, D., Anderson, C., Domingos, P., Etzioni, O., Lau, T., Gajos, K. & Wolfman, S. (2003). Automatically personalizing user interfaces. In Proceedings IJCAI'03. San Francisco: Morgan Kaufmann, 1613-1619.
35. Wobbrock, J. O., Kane, S. K., Gajos, K. Z., Harada, S., & Froehlich, J. (2011). Ability-based design: Concept, principles and examples. ACM Transactions on Accessible Computing, Vol. 3, No. 3, Article 9.